Name: Aadhi Sivakumar

NetID: axs220444

Section: CS3345.002

Date: 09/22/2024

Java Stack Implementation

ArrayStack:

In the ArrayStack implementation, the stack is implemented using a dynamically resized array. The ArrayStack utilizes an internal array (stackElements[]) of type double to store the stack elements. It also uses an integer variable called size which is initialized in the start to 0 to track the number of elements that are currently in the stack and as the index for the next available position in the array. If the array reaches its capacity during a push() operation, it calls the function resizeArray(). This function creates a new array and copies the elements from the old array to the new one. It also doubles the capacity of the array to take in additional elements.

Key Parts:

- isEmpty() Method: This method is a boolean method which returns true if the stack has no elements, size = 0, or false if size != 0.
    - Average Case: O(1)
    - Worst Case: O(1)
- count() Method: This method returns the number of elements currently in the stack.
    - Average Case: O(1)
    - Worst Case: O(1)

- push(double d) Method: This method checks to see if the stack is full using (if size == stackElements.length). If it is, it calls the resizeArray() method, doubling the capacity of the internal array. It then adds a new element to the stack by this line (stackElements[size++] = d).
    - Average Case: O(1) [when array hasn't reached capacity]
    - Worst Cse: O(N) [when array has reached capacity and calls resizeArray()]
- pop() Method: This method first checks if the stack is empty using the isEmpty function in an if statement. If it is empty, it throws an EmptyStackException(). It then removes the top element of the stack, returning its value. The size is decremented but the size of the array remains the same.
    - Average Case: O(1)
    - Worst Case: O(1)
- peek() Method: This method also uses an if statement with the isEmpty function to check if the stack is empty and throws an EmptyStackException() if it is. It then retrieves the top element without modifying the stack.
    - Average Case: O(1)
    - Worst Case: O(1)
- resizeArray(): This method is utilized when the array is full and copies the elements of the old array to the new array while doubling its capacity. I used System.arraycopy to transfer the elements in the old array to the new array.
    - Average Case: O(N)
    - Worst Case: O(N)

ListStack:

In the ListStack implementation, the stack is implemented using a linked list. Each node in the stack contains a value and a reference to the next node in the list. This implementation implements all the BKStack and Iterable interface methods. It also uses the ListStackNode and ListIterator class.

Key Parts:

- Initialization: The ListStack class is initialized with the topNode(head) reference set to null, indicating that the stack is empty. The size variable is set to 0 since there are no elements in the stack.
- isEmpty() Method: This method is a boolean method which returns true if the stack has no elements, size = 0, or false if size != 0.
  - Average Case: O(1)
  - Worst Case: O(1)
- count() Method: This method returns the number of elements currently in the stack using an enhanced for loop where a count variable is incremented.
  - Average Case: O(N)
  - Worst Case: O(N)
- push(double d) Method: This method creates a new node through the line (ListStackNode newNode = new ListStackNode(d)). It then links the new node to the previous top node by setting the (newNode.next = this.topNode). It then assigns the top node to the new node (this.topNode = newNode) and increments the stack size. It also increments the modCount.
  - Average Case: O(1)
  - Worst Case: O(1)

- pop() Method: This method first checks if the stack is empty using an if statement with the isEmpty() method. If it is, it throws the EmptyStackException. It then gets the value of the top element (double poppedValue = this.topNode.getData()), moves the top node reference to the next node (this.topNode = this.topNode.next), decrements the stack size (this.size--), increments modCount, and returns the popped element.
    - Best Case: O(1)
    - Worst Case: O(1)
- peek() Method: This method also uses an if statement with the isEmpty function to check if the stack is empty and throws an EmptyStackException() if it is. It then retrieves the top element without modifying the stack.
    - Best Case: O(1)
    - Worst Case: O(1)
- Iterator Implementation: The ListStack class implements the Iterable<Double> Interface, allowing traversal of the elements in the stack. This iterator provides:
    - hasNext() Method: Checks if there are more elements to iterate.
        - Average Case: O(1)
        - Worst Case: O(1)
    - next() Method: Retrieves the next element, throwing a NoSuchElementException() if there are no more elements.
        - Average Case: O(1)
        - Worst Case: O(1)
    - checkForConcurrentModification() Method: The iterator checks throws a ConcurrentModificationException if the stack is modified during iteration.